Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.24, NO.5, Oct. 2014 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2014.24.5.941

클라우드 컴퓨팅 환경에서의 개인정보보호를 위한 완전 동형 암호 적용 방안 고찰

김 세 환,† 윤 현 수‡
한국과학기술원 전산학과

A Survey of applying Fully Homomorphic Encryption in the Cloud system

Sehwan Kim,† Hyunsoo Yoon‡
Korea Advanced Institute of Science and Technology
Department of Computer Science

요 약

클라우드 컴퓨팅 서비스는 스마트기기의 보급과 맞물려 편리함을 장점으로 수요가 급격히 증가하고 있다. 시장의 관심을 받으면서 클라우드 컴퓨팅 시스템이 정말로 안전한지에 대한 관심도 높아지고 있는 상황이다. 클라우드 환경의 특성상 서비스 제공자는 데이터를 위탁받는 입장에서 사용자의 개인정보를 유출할 수 있게 되는데, 이러한 문제점을 해결하기 위한 방법으로 완전 동형 암호가 대두되고 있다. 완전 동형 암호는 사용자의 암호화된 데이터를 복호화하지 않고 연산을 가능하게 하는 암호체계이다. 완전 동형 암호를 이용하면 기밀성을 보장하면서 암호문에 대한 키가 없어도 연산을 수행할 수 있기 때문에 클라우드 컴퓨팅 환경에서 발생할 수 있는 보안 위협 요소들을 제거하는데 효과적일 것으로 전망하고 있다. 본 논문에서는 클라우드 컴퓨팅 서비스에서 발생할 수 있는 보안 위협 요소들을 조사하고 이를 해결할 수 있는 완전 동형 암호에 대하여 살펴본다.

ABSTRACT

Demands for cloud computing service rapidly increased along with the expansion of supplying smart devices. Interest in cloud system has led to the question whether it is really safe. Due to the nature of cloud system, cloud service provider can get a user's private information and disclose it. There is a large range of opinion on this issue and recently many researchers are looking into fully homomorphic encryption as a solution for this problem. Fully homomorphic encryption can permit arbitrary computation on encrypted data. Many security threats will disappear by using fully homomorphic encryption, because fully homomorphic encryption keeps the confidentiality. In this paper, we research possible security threats in cloud computing service and study on the application method of fully homomorphic encryption for cloud computing system.

Keywords: Fully homomorphic encryption, Cloud computing

I. 서 론

컴퓨터 성능의 발전과 스마트기기의 등장으로 이동성이 용이해지면서 컴퓨팅 환경이 변화하고 있다. 모든 업무의 온라인 처리가 가능하게 되어, 기존의 장비를 갖춰야만 할 수 있었던 여러 가지 작업들은 장비가 없어도 클라우드 컴퓨팅 서비스(이하 클라우드 서비스)를 이용하여 해낼 수 있게 되었다. 이동식 드라이브를 들고 다니지 않아도 스토리지 서비스를 이용하여 어디서든 데이터에 접근이 가능해졌으며,

접수일(2014년 6월 16일), 수정일(2014년 9월 2일), 게재확정일(2014년 10월 6일)
† 주저자, shkim@nslab.kaist.ac.kr
‡ 교신저자, hyoon@kaist.ac.kr(Corresponding author)

사용자의 컴퓨터 성능 부족으로 처리할 수 없었던 연산 작업을 위탁 처리해주는 서비스까지 등장하게 되었다. 사용자는 클라우드 서비스 공급자에게 일정 금액만 지불하면 원하는 작업을 진행하면서 장비 구매, 소프트웨어 구매, 유지보수 비용을 크게 절감할 수 있게 되었다[1].

이러한 장점들을 가진 클라우드 서비스는 시장이 계속 팽창되고 있는 상태이며, 스마트기기 보급률이 점점 높아지는 현재 환경에서 앞으로의 발전 가능성이 더 기대되는 서비스이다. 조사기관 IDC에 따르면 국내 퍼블릭 클라우드 서비스 시장이 2013년 4억 8700만 달러에서 2017년 12억 달러 규모로 약 3배 정도 시장이 팽창할 것으로 전망하고 있다[2]. 하지만 시장 조사와는 달리 실제 클라우드 서비스를 이용할 기업들은 안전성에 대해 의심을 가지고 있다. 2013년 마이크로소프트에서 발표한 설문조사에 따르면 설문조사 대상 국내 기업 234곳 중 82%가 시장의 낮은 이해도로 어려움을 겪는 것으로 나타났는데 대부분 클라우드 서비스의 보안 수준에 대해 우려를 표명했다[3]. 기업의 입장에서는 클라우드 서비스를 이용하면서 사내 인프라 관리의 부담은 덜 수 있으나 데이터를 위탁하는 방식은 기업의 기밀자료가 제 3자의 서버에 저장되게 되어 보안상 클라우드 서비스의 이용을 기피하게 된다는 것이다.

단순한 스토리지 서비스만 이용한다면 기존 암호화 알고리즘으로 암호화된 파일을 업로드하면 되기 때문에 클라우드 서비스 제공자 측에서는 복호화를 할 필요가 없다. 하지만, 사용자가 클라우드 서버 내에서 파일의 검색이나 연산을 요청하는 경우에는 클라우드 서비스 제공자는 암호화된 파일로는 연산을 진행 할 수 없다. 암호화되지 않은 평문 파일을 제공받거나 암호화된 파일을 풀 수 있는 암호키가 있어야 하는데, 이 경우 제 3자가 평문 정보를 알게 되어 정보유출의 소지가 있다. 이런 문제점을 보완하기 위하여 암호화된 파일을 복호화하지 않고 검색할 수 있는 검색가능 암호시스템 등이 연구 되고 있으며 최근 완전 동형 암호기법이 대두되고 있다.

완전 동형 암호(Fully Homomorphic Encryption, FHE)는 사용자의 암호화된 데이터를 복호화하지 않고 연산을 가능하게 하는 암호체계이다[7]. 암호문을 복호화하지 않은 상태로 연산을 하더라도 그에 따른 결과가 평문에 대한 연산과 대응되기 때문에 암호문을 가지고 있는 제 3자에게 평문이나 암호키를 전달하지 않아도 연산을 의뢰할 수 있게 된다. 따라서 클라우드 서비스에서 제기된 제 3자에 의한 정보유출을 완전 동형 암호를 이용하여 방지할 수 있게 된다. 1978년 처음 가능성이 제시된 완전 동형 암호는 이후 안전성을 보장하지 못하고 오랜 시간 기술이 정체되어 있다가 2009년 Gentry[4]에 의해 새로운 개념이 제시되었고 이를 토대로 연구가 발전되고 있다.

본 논문의 구성은 다음과 같다. 2장에서 클라우드 서비스의 보안 체계를 위협할 요소를 분석하고 3장에서 대응책으로 제시될 완전 동형 암호의 정의를 살펴보고 동향을 조사한다. 4장에서는 실제 구현을 통하여 완전 동형 암호의 성능을 분석하고, 5장에서는 클라우드 서비스에 완전 동형 암호를 적용시킬 시나리오를 설계하며 완전 동형 암호를 적용할 때 생길 수 있는 문제점을 제시한다. 6장에서 결론을 맺는다.

II. 클라우드 서비스 보안 요소

클라우드 서비스 시스템의 객체는 크게 3개의 유형으로 분류될 수 있다. 첫 번째는, 데이터의 실제 주인으로 서비스 사용자이다. 두 번째는, 데이터를 저장하는 클라우드 서비스 공급자이며 마지막은 클라우드 서비스 사용자이나 데이터의 주인은 아닌 제3자이다. 사용자가 기존 클라우드 시스템이 아닌 서버를 직접 사용할 때에는 외부 침입자만 보안 요소로 고려했었다. 하지만 클라우드 시스템은 클라우드 서버가 데이터를 위탁 저장하기 때문에 클라우드 서비스 제공자의 침입도 고려되어야 한다. 본 장에서는 객체 별로 클라우드 서비스를 이용했을 때 침입 받을 수 있는 시나리오를 구성해본다.

2.1 클라우드 서비스 제공자의 악의적 침입

클라우드 서비스 사용자는 클라우드 서비스 제공업체가 보안상 안전하다는 판단 하에 서비스를 이용한다. 최근에는 클라우드 서버에 데이터를 단지 저장만 하는 것이 아니라 각종 연산, 작업 처리를 위탁한다. 클라우드 서버는 처음부터 사용자의 데이터 평문을 제공받을 수도 있고, 사용자가 클라우드 서버에 암호화된 데이터를 올리는 동시에 클라우드 서비스 제공자만 열어볼 수 있도록 암호키를 제공해줄 수도 있다. 하지만 두 가지 방법 모두 클라우드 서비스 제공자에게 평문이 공개되는데, 외부의 침입으로부터 안전하다 하더라도 서비스 제공자가 악의적으로 사용

자의 데이터를 유출 할 수 있게 되어 이는 결국 안전한 시스템이라고 할 수 없게 된다.

2.2 클라우드 서비스 미숙 운영과 외부의 침입

클라우드 서비스는 사용자 각각의 서버를 두는 것이 아니라 하나의 시스템에서 모든 사용자를 관리하는 방식으로 구성된다. 클라우드 서비스 제공의 미숙으로 데이터 격리가 잘 이루어지지 않는 경우를 고려했을 때 클라우드 제공자가 암호키나 평문을 저장하고 있는 것은 매우 위험하다[1]. 또한, 클라우드 서비스에 가입만 하면 클라우드 서버에 접근이 가능해지는데 외부 침입자는 이를 고려하여 클라우드 서비스에 가입을 하여 쉽게 서버에 접근할 수 있는 권한을 얻고 서버를 악의적으로 공격하여 다른 사용자들의 개인정보를 얻어낼 수 있다.

III. 완전 동형 암호의 소개와 동향

3.1 동형 암호의 개요

암호화 기법은 공개된 채널을 통하여 안전한 통신을 가능하게 하는 것으로, 도청자를 비롯한 권한이 없는 제 3자로부터 데이터의 기밀성 보장을 목표로 한다. 나아가 암호화된 상태에서 연산을 수행할 수 있는 방안에 대한 의문이 제기 되었다. 이 때문에 암호문에 대한 연산이 평문에 대한 연산으로 대응이 될 수 있는 특성을 가진 동형 암호에 대해 연구가 이루어졌으며 Rivest, Adleman, Dertouzos[5]에 의해 1978년 처음 제시되었으나 안전성의 문제로 사용되지 못했다. 이후 제한된 범위에서만 동형성이 보장되는 상태로 연산이 가능한 부분 동형 암호(somewhat homomorphic encryption)를 거쳐 2009년 Gentry에 의해 제시된 암호문에 대한 임의의 연산을 가능하게 하는 완전 동형 암호로 발전하였다. 근래에 제시된 완전 동형 암호는 대부분 Gentry에 의하여 제시된 방법을 따르고 있으며, 개념적으로 단순화, 효율성 향상 및 기반 가정을 개선하는 형태로 지속적으로 연구되고 있다.

3.2 동형 암호의 정의

보안 매개변수를 λ(lambda)라 할 때, 비트 단위 연산을 지원하는 공개키 기반의 동형 암호화에 대한 정의 HE는 확률적으로 다항식 시간 내에 계산할 수 있는 다음의 네 가지 알고리즘으로 이루어진다.

3.2.1 $HE.KeyGen$

키 생성 알고리즘으로 보안 매개변수를 입력으로 받아 복호화에 사용되는 비밀키 sk, 암호문에 대한 동형 연산에 사용되는 연산키 evk, 암호화에 사용되는 공개키 pk를 생성한다.

3.2.2 $HE.Enc$

암호화 알고리즘으로, {0,1}의 원소 μ를 공개키 pk를 이용하여 암호화 하여 암호문 c를 생성한다.

3.2.3 $HE.Dec$

복호화 알고리즘으로, 암호문 c와 비밀키 sk를 입력으로 받아, 이에 해당하는 평문 μ^*를 복원한다.

3.2.4 $HE.Eval$

동형 연산 알고리즘으로, l-비트 길이로 이루어진 암호문 c의 각각의 비트에 대한 암호문 $\{c_i\}_{1\le i\le l}$과 연산 함수 f, 연산키 evk를 입력으로 받아 암호문 c에 대한 동형 연산을 수행한다. 여기서 연산 함수 f에 대한 구현은 다양한 방법으로 이루어질 수 있으며 이에 따라 연산의 복잡도가 달라질 수 있다.

3.3 완전 동형 암호

동형 암호에는 한 가지 중요한 특징이 있다. 암호문에 임의 크기의 노이즈를 삽입한다. 문제는 이 노이즈가 연산을 거듭할수록 증가하게 되는데, 일정 수준을 넘어서면 평문의 정보가 변형이 될 만큼 노이즈가 증가하게 된다. 이렇게 연산 횟수에 제한이 생기는 동형 암호를 부분 동형 암호(somewhat homomorphic encryption)라고 한다.

완전 동형 암호는 부분 동형 암호에서 제기되었던 노이즈 증가에 따른 평문 변형의 문제점을 해결한 암호이다. 증가하는 노이즈를 조절하여 원하는 횟수만큼의 임의의 연산에 대하여 동형성이 보장되는 암호이다.

이 외에 계층적 완전 동형 암호(leveled fully

homomorphic encryption)가 있다. $HE.KeyGen$ 알고리즘이 부가적인 입력 1^L을 입력받는데 이는 완전 동형 암호처럼 임의의 횟수에 대한 연산은 아니지만, 깊이 L만큼의 연산 서킷에 대하여 동형성을 유지할 수 있는 암호이다.

3.4 완전 동형 암호의 특징

3.4.1 간결성

$HE.Eval$에서 동형 연산의 결과로 생성되는 암호문의 길이가 입력 길이인 l또는 연산 함수 f의 시간 복잡도에 영향을 받지 않았을 때 간결성을 갖추었다고 한다. 즉, 암호문의 길이는 f의 연산 결과의 크기에 대해서만 영향을 받는다.

3.4.2 서킷 프라이버시(circuit privacy)

동형 연산 f의 정보는 비밀키를 알지 못하고 연산을 수행하는 자에 의하여 수행되는데, 연산을 진행하는데 있어 연산 f에 대한 정보를 알 수 없는 경우 서킷 프라이버시가 보장된다고 한다. 따라서 연산 요청자는 동형 연산의 결과로부터 f에 대한 정보를 유추할 수 없도록 해야 함을 의미한다.

3.4.3 다중 도약 동형성

$HE.Eval$의 결과로 생성된 암호문이 다른 동형 연산의 입력으로 사용될 수 있는 경우, 이를 다중 도약 동형성이 보장되는 동형 암호화라 하며, Gentry, Halevi, Vaikuntanathan[6]에 의해 개념이 제시되었다.

3.5 Gentry가 제안한 완전 동형 암호

Gentry는 2009년 특정 조건을 만족시키는 부분 동형 암호로부터 완전 동형 암호를 구성할 수 있는 방법을 제안하였다[4]. 부분 동형 암호를 먼저 구성하고, 동형성이 유지되는 상태에서 완전 동형 암호화를 구성할 수 있는 방법을 제시하였는데, 이 과정에서 부트스트래핑(bootstrapping)과 스쿼싱(squashi-ng)기법이 중요하게 다뤄진다.

3.5.1 부트스트래핑(bootstrapping)

동형 암호에서 완전 동형 암호로 발전하기 위한 중요한 요소 중 하나가 노이즈 조절이다. 암호화로부터 증폭되는 노이즈를 평문이 변형되지 않을 정도로 조절할 수 있다면 원하는 횟수만큼의 동형 연산을 진행할 수 있다. 부트스트래핑은 암호문의 암호화된 결과로부터 암호화된 비밀키를 이용하여 노이즈가 감소된 새로운 암호문을 생성한 후, 덧셈 및 곱셈 등의 연산을 수행하는 과정을 의미한다. 이 과정을 반복하면 새로운 암호문을 생성할 때마다 노이즈가 감소되기 때문에 완전 동형 암호를 구성할 수 있다.

3.5.2 스쿼싱(squashing)

동형 복호화 서킷을 어떻게 구성하는가에 따라 연산 과정에서의 노이즈 증가로 인해 원하는 결과를 얻지 못하는 경우가 발생한다. 따라서 동형 복호화 서킷은 노이즈 증가를 줄이고 평문이 변형되지 않도록 구성되어야 하는데 이를 스쿼싱이라고 한다. 처음 완전 동형 암호 알고리즘을 구체적으로 제시한 Gentry는 [4]에서 스쿼싱을 위해 SSSP(Sparse Subset-Sum Problem)을 어려운 문제라고 하고 이용하며 이후에 제시된 논문들도 다수 이 방식을 따르고 있다.

3.6 동형 암호의 기반에 따른 분류

완전 동형 암호는 알고리즘의 연산 기반에 따라 분류를 할 수 있다. 크게 4가지로 나뉘는데 Gentry가 처음 제안했던 개념은 격자기반의 알고리즘이었다. 이외에도 LWE(Learning With Error), Ring-LWE, NTRU 기반이 있으며 정수기반의 알고리즘이 있다. 최근에는 효율성에 있어 Ring-LWE 기반의 연구와 상대적으로 이해하기 쉬운 정수 기반의 연구가 활발하게 진행되고 있다.

3.7 Fully Homomorphic Encrypthon over the Integers(DGHV, Eurocrypt 2010)

최근 완전 동형 암호는 상대적으로 이해가 쉬운 정수를 기반으로 한 연구가 활발하게 진행되고 있다. 본절에서는 정수 기반 완전 동형 암호 연구의 기초가 되는 논문을 분석하고자 한다. 이 논문은 Gentry의

2009년 초기 알고리즘을 바탕으로 2010년 van Dijk, Gentry, Halevi, Vaikuntanathan[9]에 의해 제안되었으며 일반적으로 DGHV라 불리고 있다. 정수 기반으로 기본적인 모듈러 연산(modular arithmetic)만을 이용하면서 상대적으로 단순한 개념이 특징이다. 0 또는 1의 값을 가지는 비트 수준의 평문에 대하여 XOR 연산과 AND 연산을 지원하는 정수 상에서의 완전 동형 암호화 기법을 제시함으로써 이를 확장한 정수에서의 연산이 가능함을 보이고 있다.

3.7.1 매개변수

이들이 제시한 방법에서 사용되는 매개변수를 살펴보면 아래의 표와 같다.

Table 1. arguments of DGHV

λ	security parameter
γ	bit length of public key
η	bit length of secret key
ρ	bit length of noise
ρ'	bit length of newly noise
τ	the number of elements of public key

3.5절에서 살펴본 것과 같이 완전 동형 암호를 위해 DGHV에서도 스퀴싱을 하는데 이 때 사용되는 매개변수는 아래의 표와 같다.

Table 2. arguments for squashing

κ	$\gamma\eta/\rho'$
θ	λ
Θ	$\kappa \bullet \log\lambda$ *is lower boundary*

3.7.2 표기법 정의

실수 z를 정수로 변환하는 과정에서 다음과 같은 정의가 사용된다.

Table 3. notation about rounding

$\lceil z \rceil$	rounding of z up
$\lfloor z \rfloor$	rounding of z down
$\lfloor z \rceil$	rounding of z to the nearest integer

모듈러 연산에 대한 표기법은 다음과 같다.

Table 4. notation about modular

$[x]_y$	$x \bmod y$

3.8 DGHV의 알고리즘 구성

DGHV도 앞서 살펴본 동형 암호의 알고리즘 구성과 같으며 내용은 다음과 같다.

3.8.1 *DGHV.KeyGen*

임의의 η-비트 홀수 p를 선택하여 secret key sk^*로 설정한다. $i=0,...,\tau$에 대하여 임의의 x_i를 얻어낸다. 여기서 x_i는 공개키의 요소로 사용되는데 $x_i = q_i \bullet p + r_i$로 나타낼 수 있으며, q_i는 $[0,2^\gamma/p)$의 범위를 가지는 정수이고 r_i는 $(-2^\rho, 2^\rho)$의 범위를 가지는 정수이다. 이를 $pk^* = < x_0,...,x_\tau >$로 구성하며 가장 큰 x_i를 x_0로 설정한다. 1의 개수가 θ인 임의의 Θ-비트 벡터 $s = < s_1,...,s_\Theta >$를 만들고 성긴 부분집합 (sparse subset) $S = \{i : s_i = 1\} \subset \{1,...\Theta\}$을 구성한다.

$i=1,...,\Theta$에 대하여 $\sum_{i\in S} u_i = x_p$를 만족하는 임의의 정수 u_i를 선택한다.(단, $x_p \leftarrow \lfloor 2^\kappa/p \rceil$ 이며, $p = sk^*$) 그리고 κ-비트의 정확도를 가지는 $y_i = u_i/2^\kappa$를 계산하여 $y = \{y_1,...,y_\Theta\}$를 구성한다. 비밀키를 $sk = s$로 설정하고, 공개키 $pk = (pk^*, y)$를 배포한다.

3.8.2 *DGHV.Encrypt*

임의의 부분집합 $S \subseteq \{1,...,\tau\}$를 선택하고 노이즈로 사용할 정수 r을 $(-2^{\rho'}, 2^{\rho'})$ 범위에서 선택한다. 평문 $m \in \{0,1\}$에 대하여, 암호문 c^*를 다음과 같이 계산한다. $c^* \leftarrow [m + 2r + 2\sum_{i\in S} x_i]_{x_0}$

$i=1,...,\Theta$에 대하여 $z_i \leftarrow [c^* \bullet y_i]_2$를 각각 계산하여 $c = (c^*, z = < z_1,...z_\Theta >)$를 결과로 반환한다.

3.8.3 *DGHV.Decrypt*

비밀키 sk와 암호문 c에 대하여 다음과 같은 연산을 수행하여 평문을 얻는다.

$m' \leftarrow [c^* \lfloor \sum_i s_i \bullet z_i \rceil]_2 \approx [c^* - \lfloor c^* \bullet 1/p \rceil]_2$

3.8.4 *DGHV.Evaluate*

암호화된 상태에서 연산을 수행하기 위해 *Evaluate* 서킷이 암호 기법에 포함이 된다. 암호문 $c=<c_1,...,c_t>$에 대하여, t개의 입력 c_i을 취하는 서킷 C_e가 주어졌을 때, 서킷의 곱셈 및 덧셈에 대한 게이트를 암호문에 적용하여 정수 상에서의 연산을 수행하고 결과로 생성되는 정수를 반환한다. 여기서 *Evaluate* 서킷은 수행 결과로 생성되는 암호문으로부터 해당 서킷이 어떠한 연산을 수행하는지에 대한 정보를 노출시키지 않아야 한다는 서킷 프라이버시를 만족해야 한다.

IV. 완전 동형 암호 알고리즘 성능 분석

4.1 실험 환경

실험은 intel i3-3220 CPU @ 3.30GHz, 8GB RAM의 성능을 가진 컴퓨터에서 진행되었다. 이러한 환경은 요즘 대부분 가정집이나 사무실에 보급된 컴퓨터 사양으로 완전 동형 암호를 실제 상황에서 사용할 수 있는지 알아보고자 하는 목적이 있다. 다음 장에서 제시될 클라우드 환경에서의 완전 동형 암호 사용 시나리오에서 데이터를 암호화 하는 주체가 클라우드 서버가 아닌 사용자이기 때문에 실제 사용자라 예상할 수 있는 컴퓨터의 환경에 맞게 실험 환경을 구축하였다.

4.2 구현 내역

[9]의 공개키 방식 완전 동형 암호 알고리즘을 구현하였다. 구현은 C++를 통해 이루어졌으며, 큰 정수 연산 처리가 가능한 GMP library 5.1.3을 이용하였다.

알고리즘의 매개변수는 [10]의 실험에서 사용한 매개변수를 이용하였다. 1비트의 평문을 암호화 하여 단계별로 실험을 진행하였다. 각 단계의 소요시간을 분석하였고 결과는 Table 5, 와 같다.

Table 5, Result of experiment

(unit : sec)

Parameter	*KeyGen*	*Encrypt*	*Add*	*Mul*
$\lambda=42$	0.07	0.00	0.00	0.00
$\lambda=52$	0.88	0.01	0.00	0.02
$\lambda=62$	12.76	0.024	0.01	0.16

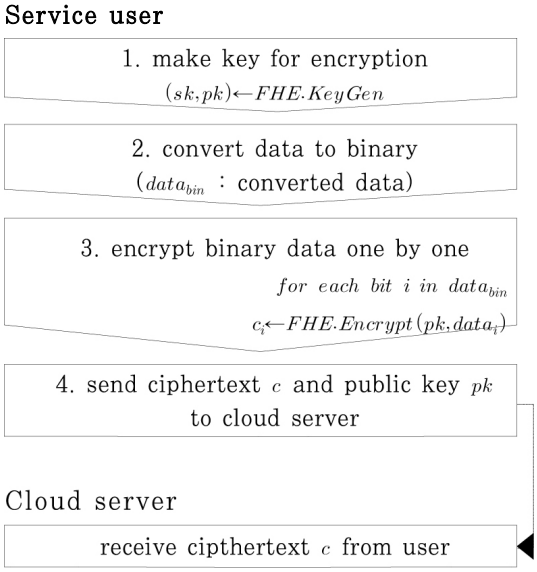
Parameter	*Expand*	*Recrypt*	*Decrypt*
$\lambda=42$	0.08	0.16	0.00
$\lambda=52$	2.31	1.33	0.00
$\lambda=62$	75.15	13.20	0.00

V. 클라우드 환경에서의 완전 동형 암호 적용

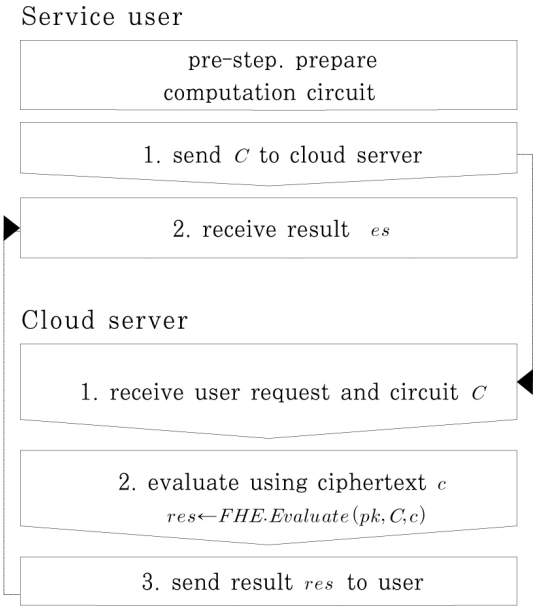
5.1 완전 동형 암호 적용 시나리오

2장에서 살펴본 것과 같이 클라우드 서비스에 있어 보안 측면에서 가장 큰 문제점은 클라우드 서비스 제공자가 평문 정보를 알 수 있다는 것이었다. 이전까지의 암호화 알고리즘으로는 암호화된 데이터를 연산에 이용하기 위해서는 복호화가 반드시 필요했다. 하지만 완전 동형 암호를 이용하면 암호화된 데이터를 복호화 없이 연산하더라도 그 결과가 평문에서의 연산 내용과 대응되기 때문에 클라우드 서비스 제공자는 위탁받은 내용을 평문이나 키 정보 없이 처리하여 사용자에게 전달할 수 있고 사용자는 자신이 가진 키로 복호화하여 연산된 결과를 받을 수 있다. 이 경우 클라우드 서버가 평문을 모르게 하기 위해 평문의 암호화나 키 생성은 사용자 측에서 진행되어야 하며, 사용자가 완전 동형 암호 알고리즘을 작동시킬 수 있다는 가정 하에 적용 될 수 있는 시나리오는 다음과 같다.

5.1.1 클라우드 서버 업로드 시나리오



5.1.2 사용자의 연산 요청 시나리오



5.2 완전 동형 암호 적용시의 보안성 검토

완전 동형 암호의 적용은 사용자가 평문을 배포하지 않아 데이터 기밀성을 보장한다. 특히 클라우드 환경에서 보호가 필요한 개인정보의 유출을 막는 방법이 된다. 하지만 현재 단계에서는 클라우드 시스템에 완전 동형 암호를 적용시키기에 몇 가지 문제점이 발생한다.

5.2.1 완전 동형 암호의 효율성

완전 동형 암호는 기능적으로는 혁신적이고 암호화된 상태에서 다양한 연산을 가능하게 함으로써 다양한 응용기반을 제공하고 있지만, 심각한 효율성의 문제를 포함하고 있다. 3.6절에서 살펴보았던 DGHV의 경우 정수기반으로 비트 단위의 암호화를 하기 때문에 암호화를 할 데이터를 모두 비트단위로 바꾸어 연산을 해야하는 불편함이 있으며 공개키의 크기가 $O(\lambda^{10})$로 2^{60}에 달하게 된다. DGHV가 발표된 이후 최근까지도 연구를 통하여 공개키의 크기를 줄이고는 있으나 아직 실용적이지는 못한 수준이다. 정수 기반의 완전 동형 암호의 경우는 2에 대한 모듈러 연산이 알고리즘의 기반이 된다. 이를 해결하기 위해서는 2 보다 더 큰 수의 모듈러 연산을 통해 암호화 알고리즘을 구성할 수 있는 방법이 요구된다. 이를 통해 동일한 성능으로 보다 많은 정보의 암호화를 가능하게 할 수 있다.

암호화에 요구되는 시간도 문제가 된다. 2012년 DGHV를 개선한 정수기반의 완전 동형 암호화 알고리즘의 경우 공개키의 크기는 10MB 정도로 크게 감소시켰으나 암호화 과정이나 노이즈를 조절하는 과정에서 실용적으로 알고리즘을 사용할 수 있는 수준의 성능을 보이지 못한다[10].

4절의 성능 분석을 통해 살펴보면 Medium 수준의 보안 매개변수를 사용할 경우 과정에서 약 1분이 넘는 소요 시간을 보여주고 있다. 위의 성능 분석에서 언급되지는 않았으나 실험 결과 암호문의 노이즈의 크기가 약 1번 정도의 곱셈 연산이 가능할 정도로 측정되었다. 실험 분석 결과 곱셈 연산을 한번 할 때마다 노이즈를 줄이는 과정인 부트스트래핑이 요구되며 이를 위해 $Expand$와 $Recrypt$가 호출된다. 이 두 과정은 소요시간 중 가장 많은 비중을 차지하는 과정이며 1비트의 평문을 암호화하기에는 다소 많은 시간이 소요된다. 적어도 KB, 많게는 MB, GB까지 크기를 가진 문서가 클라우드 서버에 올라오는 것이 현재 상황이기 때문에 암호화 과정의 소요 시간을 단축해야 하는 문제를 안고 있다. 최근 Coron, 천정희 등[11]이 제안한 완전 동형 암호는 이를 해결 할 수 있는 새로운 방향을 제시해 주고 있다. 비트단위의 암호문을 일괄 처리하여 암호화에 요구되는 시간을 단축하고 있다.

5.2.2 데이터의 무결성 보장

동형 암호는 암호문에 대한 연산이 복호화 없이 이루어질 수 있다는 특징이 있다. 이는 장점이 될 수도 있으나 동시에 단점이 될 수도 있다. 기존 암호문은 연산을 해도 전혀 사용이 불가능한 정보를 얻게 되지만 동형 암호의 암호문은 사용이 가능한 정보를 얻을 수가 있다. 예를 들어, 개인의 월급정보를 동형 암호 알고리즘으로 암호화하여 저장하면 같은 암호문을 제곱을 한 암호문은 복호화를 했을 때 개인의 월급이 제곱이 되어 나타나게 된다. 즉, 동형 암호화는 비밀키가 없더라도 암호문에 대한 연산을 가능하다. 공격자는 암호문을 무작위로 연산하면서 평문을 변형시켜 무결성을 해칠 수 있게 된다.

5.2.3 데이터의 기밀성 보장

완전 동형 암호를 공격하는 방식은 두가지 정도로 나눌 수 있다. 첫 번째는 비밀키 p를 알아내는 공격이며, 두 번째는 암호문으로부터 평문 m을 알아내는 방법이다[12]. 정수 기반의 완전 동형 암호 알고리즘은 무차별 공격에 대응할 수 있도록 보안 매개변수의 크기를 설정하고 있다[9]. 하지만, 격자 축소 알고리즘을 이용하여 공격이 가능한 것으로 알려져 있다. 특히 데이터의 기밀성을 보장하기 위해 완전 동형 암호 알고리즘에서는 암호문에 노이즈를 더하게 되지만 이 노이즈가 암호문을 공격하는 방안이 될 수 있다. 노이즈의 크기는 다른 매개변수보다 상대적으로 크기가 작기 때문에 기밀성 보장의 약점이 된다. 최근 격자 축소 알고리즘을 이용한 완전 동형 알고리즘의 공격은 [9]에서 설정하였던 보안수준을 맞추지는 못하였으나 보안 매개변수 $\lambda = 12$인 경우 비밀키 p를 알아내는데 약 7063초가 걸리는 것으로 실험결과가 나타났다[12].

VI. 결 론

클라우드 컴퓨팅 서비스는 개인의 스마트기기 보급 확산과 더불어 폭발적으로 성장하고 있으며 사용자들은 보다 편리한 환경을 제공받게 되었다. 하지만 기능적 측면의 성장에 비해 보안적인 측면에서 클라우드 컴퓨팅 서비스는 아직 보완해야할 점들이 많다. 기존의 서버환경과 다르게 데이터를 위탁하는 방식이기 때문에 사용자가 클라우드 컴퓨팅 서비스의 사용을 꺼리게 하는 장벽이 되고 있으며 이로 인해 클라우드 컴퓨팅 서비스의 활성화가 더더지고 있다.

본 논문에서는 클라우드 컴퓨팅 보안의 해결책으로 제시되고 있는 완전 동형 암호를 살펴보았다. 완전 동형 암호는 기능에 있어 클라우드 컴퓨팅 보안의 핵심이 될 수 있는 암호 체계이지만, 효율성이 문제가 되어 바로 시장에 투입하기에는 어렵다. 최근에는 연구를 통해 점차 개선이 되어 앞으로 다양한 분야에 응용이 될 것으로 기대하고 있으며 특히 클라우드 컴퓨팅 분야에서 큰 활약을 할 것으로 예상하고 있다. 효율성으로는 아직 부족하지만 기능적으로 개인정보 유출에 대한 해결책이 될 수 있기 때문에 앞으로 기능적인 장점은 보존하면서 효율적인 측면을 끌어올릴 수 있는 연구가 진행되어야 할 것이다.

References

[1] Kyoung-a Shin, Sang-jin Lee, "Information security management system on cloud computing service," Journal of The Korea Institute of information Security & Cryptology, 22(1), pp. 155-167. Feb. 2012.

[2] http://www.idckorea.com/product/Getdoc.asp?idx=547&field=PressRelease

[3] http://www.microsoft.com/apac/news/cloud-myths/all/index.html

[4] Gentry, Craig. "A fully homomorphic encryption scheme," Ph.D.Thesis, Stanford University, Sep. 2009.

[5] Rivest, Ronald L., Len Adleman, and Michael L. Dertouzos. "On data banks and privacy homomorphisms," Found-ations of secure computation, pp. 169-180, 1978.

[6] C. Gentry, S. Halevi, and V. Vaikuntanathan, "i-hop homomorphic encrypt-ion and rerandomizable yao circuits," CRYPTO 2010, LNCS, vol.6223, pp. 155-172, 2010.

[7] Myoung In Jeong, "Technical trend of fully homomorphic encryption," The Journal of the Korea Contents Association, 13(8), pp. 36-43, Aug, 2013.

[8] V. Vaikuntanathan, "Computing blindfolded: New developments in fully homomorphic encryption," Foundations of Computer Science, 2011 IEEE 52nd Annual Symposium on. IEEE, pp. 5-16, Oct. 2011.

[9] M. Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," Advances in Cryptology, EUROCRYPT 2010, LNCS 6110, pp. 24 - 43, 2010.

[10] JS Coron, D. Naccache, and M. Tibouchi. "Public key compression and modulus switching for fully homomorphic encryption over the integers," Advances in

Cryptology, EUROCRYPT 2012. LNCS 7237, pp. 446-464, 2012.
〔11〕 Cheon, Jung Hee, et al, "Batch fully homomorphic encryption over the integers," Advances in Cryptology, EUROCRYPT 2013, LNCS 7881, pp. 315-335, 2013.
〔12〕 Ding, Jintai, and Chengdong Tao. "A new algorithm for solving the general approximate common divisor problem and cryptanalysis of the FHE based on the GACD problem," IACR Cryptology ePrint Archive, Report 2014-042, Mar, 2014.

〈저자소개〉

김 세 환 (Sehwan Kim) 학생회원
2013년: 한국과학기술원 전산학 학사
2013년~현재: 한국과학기술원 전산학 석사과정
〈관심분야〉 정보보호

윤 현 수 (Hyunsoo Yoon) 정회원
1979년: 서울대학교 전자공학과 학사
1981년: 한국과학기술원 전산학 석사
1981년~1984년: 삼성전자 연구원
1988년: 오하이오 주립대학 전산학 박사
1988년~1989년: AT&T Bell Labs. 연구원
1989년~현재: 한국과학기술원 전산학 교수
〈관심분야〉 암호학, 상호연결 네트워크, Adhoc망, 병렬 컴퓨터